

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-319729

(43)Date of publication of application : 08.12.1995

(51)Int.Cl.

G06F 11/28

G06F 9/06

G06F 9/45

(21)Application number : 06-106420

(71)Applicant : HITACHI LTD
HITACHI COMPUT ENG CORP LTD

(22)Date of filing : 20.05.1994

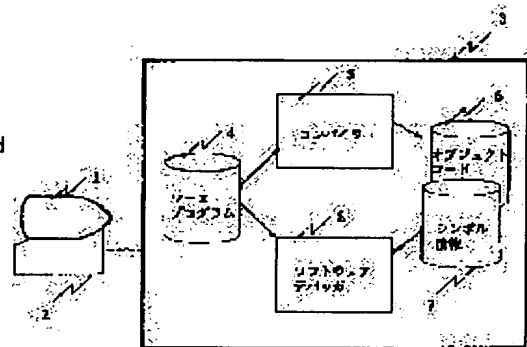
(72)Inventor : TANAKA YUJI
SASAKI TAMOTSU
KOIZUMI AKINORI

(54) SOFTWARE DEBUGGING METHOD

(57)Abstract:

PURPOSE: To verify a program to be verified in exactly the same state as the program to be built in a real product by showing a source program row to which the address of an object code is assigned and another source program row to which the address of the object code is not assigned on a display device after discriminating both rows from each other based on the inputted information.

CONSTITUTION: The information on the corresponding relation between the address of an object code 6 produced in the optimization processing step when a source program 4 to be verified is compiled and the row number of the program 4 is inputted through a keyboard 2 before the start of execution of the processing that is carried out based on the code 6 that compiled the program 4. Then the row of the program 4 to which the address of the code 6 is assigned and another row of the program 4 to which the address of the code 6 is not assigned are discriminated from each other based on the information inputted through the keyboard 2 and shown on a display device 1 to the display instruction of the program 4. Thus the program 4 can be verified with high efficiency.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平7-319729

(43) 公開日 平成7年(1995)12月8日

(51) Int.Cl.⁸

G 0 6 F 11/28

9/06

9/45

識別記号

A 7313-5B

5 4 0 S 7230-5B

7737-5B

F I

G 0 6 F 9/ 44

3 2 2 F

技術表示箇所

審査請求 未請求 請求項の数 2 O L (全 7 頁)

(21) 出願番号 特願平6-106420

(22) 出願日 平成6年(1994)5月20日

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(71) 出願人 000233011

日立コンピュータエンジニアリング株式会
社

神奈川県秦野市堀山下1番地

(72) 発明者 田中 裕二

神奈川県秦野市堀山下1番地 日立コンピ
ュータエンジニアリング株式会社内

(74) 代理人 弁理士 秋田 収喜

最終頁に続く

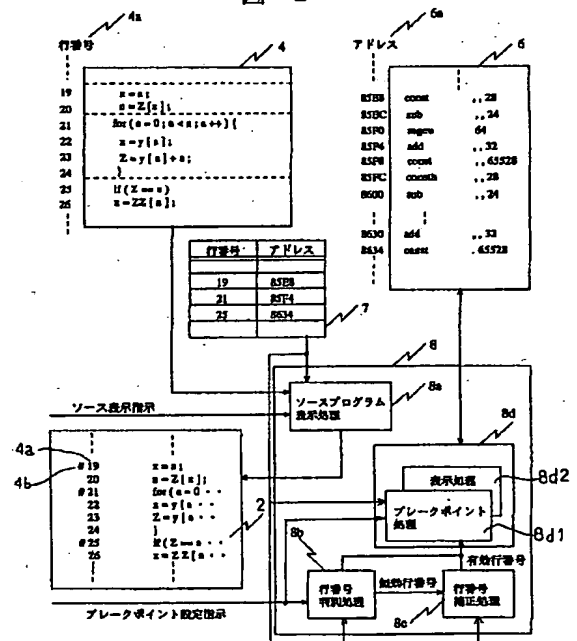
(54) 【発明の名称】 ソフトウェアデバッグ方法

(57) 【要約】

【目的】 検証対象のプログラムを実製品に組み込まれるプログラムと全く同じ状態で検証することができ、かつその検証作業を効率良く行うこと。

【構成】 検証対象のソースプログラムをコンパイルしたオブジェクトコードに従う処理の実行開始前に、検証対象のソースプログラムのコンパイル時の最適化処理段階においてオブジェクトコードのアドレスと、ソースプログラム行番号との対応関係を示すテーブルを作成する段階と、ソースプログラムの表示指示に対し、オブジェクトコードのアドレスが割り当てられたソースプログラム行と割り当てられないソースプログラム行とを前記テーブルの情報に基づいて区別して表示装置に表示させる段階とを設けたことを特徴とする。

図 2



【特許請求の範囲】

【請求項1】 検証対象のソースプログラムをコンパイル処理によってオブジェクトコードに変換し、そのオブジェクトコードに従う処理を順次実行させることによって前記ソースプログラムを検証するソフトウェアデバッグ方法において、

前記オブジェクトコードに従う処理の実行開始前に、前記検証対象のソースプログラムのコンパイル時の最適化処理段階において作成されたオブジェクトコードのアドレスと、ソースプログラム行番号との対応関係を示す情報を入力し、ソースプログラムの表示指示に対し、オブジェクトコードのアドレスが割り当てられたソースプログラム行と割り当てられないソースプログラム行とを前記情報に基づいて区別して表示装置に表示させる段階を有することを特徴とするソフトウェアデバッグ方法。

【請求項2】 前記アドレスが割り当てられないソースプログラム行での実行停止指示または実行開始指示に対しては、オブジェクトコードアドレスが割り当てられている直前のソースプログラムの行番号に補正した後、オブジェクトコードに従う処理を実行させることを特徴とする請求項1記載のソフトウェアデバッグ方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明は、ソフトウェア開発作業で開発したプログラムの検証を行うソフトウェアデバッグ方法に関するものである。

【0002】

【従来の技術】 一般に、新規に作成したソースプログラムの検証に際しては、該ソースプログラムをコンパイラによってコンパイルしてオブジェクトコードに変換し、これをデバッガに実行させることにより設計通りに動作するかどうかを確認する方法が採られる。そして、設計通りに動作することが確認されたソースプログラムは実製品に組み込まれて出荷される。

【0003】 この場合に、近年のコンパイラにあっては最適化技術が向上し、ソースプログラムの複数行を一括して最適化するため、ソースプログラムの行単位にオブジェクトコードのアドレスが対応するのではなく、複数行のソースプログラムが複数行のアドレスに対応するというコンパイル方式になっている。

【0004】 従って、このようなコンパイル方式にあっては、検証対象のソースプログラムの行とオブジェクトコードのアドレスとが1対1に対応しなくなる部分が生じる。

【0005】 しかし、このような部分が生じると、デバッグ作業において任意のソースプログラム行nまで実行した結果を検証しようとする場合に、そのソースプログラム行nに対応するオブジェクトコードのアドレスが存在しない時には、実行を停止させるオブジェクトコードのアドレス（ブレークポイント）を設定できなくなる。

【0006】 そこで、従来は、デバッグ作業段階においては、(1) UNIXのシンボリックデバッガ(sdb)に代表されるようにコンパイラの最適化処理を抑止した状態で作成したオブジェクトコードでデバッグを行うか、(2) ブレークポイントをテスト担当者が予測し、その予測したブレークポイントまで実行させる、といったデバッグ方法を採用し、このデバッグ作業において設計通りに動作することが確認されたならば、実製品に組み込むという手法を用いている。

【0007】

【発明が解決しようとする課題】 しかしながら、コンパイラの最適化処理を抑止した状態で作成したオブジェクトコードでデバッグを行った場合、設計通りに動作することが確認されたとしても、実製品に組み込まれるプログラムとは異なるプログラムをデバッグしたことになり、厳密には実製品に組み込まれるプログラムを検証したとは言えないという問題がある。

【0008】 また、テスト担当者が予測したブレークポイントまで実行させる方法を採用した場合、1回の予測で目的のブレークポイントに的中することは少ないので、何回も予測のブレークポイントを設定する操作を繰り返さなければならなくなり、作業効率が悪いという問題がある。

【0009】 本発明の目的は、検証対象のプログラムを実製品に組み込まれるプログラムと全く同じ状態で検証することができ、かつその検証作業を効率良く行うことができるソフトウェアデバッグ方法を提供することである。

【0010】

【課題を解決するための手段】 上記目的を達成するために、本発明は、基本的には、検証対象のソースプログラムをコンパイルしたオブジェクトコードに従う処理の実行開始前に、検証対象のソースプログラムのコンパイル時の最適化処理段階において作成されたオブジェクトコードのアドレスと、ソースプログラム行番号との対応関係を示す情報を入力し、ソースプログラムの表示指示に対し、オブジェクトコードのアドレスが割り当てられたソースプログラム行と割り当てられないソースプログラム行とを前記情報に基づいて区別して表示装置に表示させる段階を設けたことを特徴とする。

【0011】

【作用】 上記手段によると、オブジェクトコードのアドレスと、ソースプログラム行番号との対応関係を示すテーブルの情報に基づき、オブジェクトコードのアドレスが割り当てられたソースプログラム行についてはアドレスが割り当てられない他のソースプログラム行と区別して各行の内容が表示装置に表示される。

【0012】 従って、コンパイル時に最適化処理を実行し、ソースプログラム行とオブジェクトコードのアドレスとが1対1に対応しなくなる部分が生じたとしても、

オブジェクトコードのアドレスが割り当てられたソースプログラム行（以下、有効行と言い、アドレスが割り当てられないソースプログラム行は無効行と言う）が明確になるので、テスト担当者は有効行のみを指定して実行停止指示または実行開始指示を行うことが可能になる。

【0013】従って、最適化処理を抑止することなく、すなわち実装置に組み込むプログラムと同じ状態で検証対象プログラムを検証することができる。また、有効行を予測する必要がなくなるので、1回の指示操作のみで有効行での実行停止または実行開始を実現することができ

【0014】

【実施例】以下、本発明を図示する実施例に基づいて詳細に説明する。

【0015】図1は、本発明を実施する装置の一実施例を示すブロック構成図である。

【0016】図において、1はディスプレイ、2はキーボード、3はワークステーションであり、このワークステーション3の内部には、検証対象のソースプログラム4をコンパイルしてオブジェクトコード6を作成すると共に、最適化処理時においてオブジェクトコード6のアドレスと、ソースプログラム4の行番号との対応関係を示す情報7（以下、シンボル情報7と言う）を作成するコンパイラ5と、ソースプログラム4の検証に際して、ソースプログラム4およびオブジェクトコード6並びに前記シンボル情報7を読み込み、オブジェクトコード6に従った処理を実行すると共に、オブジェクトコード6のアドレスが割り当てられたソースプログラム行（有効行）については無効行と区別して各行の内容をディスプレイ1に表示させるソフトウェアデバッガ8が設けられている。

【0017】図2は、ソースプログラム4、オブジェクトコード6、シンボル情報7の例およびソフトウェアデバッガ8の詳細構成を示す図であり、ソースプログラム4は行番号4aが行の先頭位置に記述され、その後処理内容を示す文字情報が記述されている。

【0018】このソースプログラム4をコンパイルしたオブジェクトコード6は、命令語6bとオペランド6cとで構成され、各オブジェクトコード行には、その格納アドレスを示すアドレス情報6aが割り当てられている。

【0019】さらに、シンボル情報7を登録するテーブルには、オブジェクトコード行のアドレス情報6aと、オブジェクトコード行に対応するソースプログラム4の行番号とが対になって登録されている。

【0020】一方、ソフトウェアデバッガ8は、ソースプログラム表示処理8a、行番号判別処理8b、行番号補正処理8c、デバッグ処理8dとからなり、デバッグ処理8dはさらにブレークポイント処理8d1、変数等の表示処理8d2などから構成されている。

【0021】図3は、以上の構成においてソースプログラム4を検証する手順を示すフローチャートである。

【0022】まず、検証対象となるソースプログラム4をコンパイラ5に読み込ませ、コンパイル処理を行うが、この際に、最適化処理を行い、かつソフトウェアデバッガ8を使用するかどうかをテスト担当者からの入力情報によって判定する（ステップ30）。最適化処理を行い、かつソフトウェアデバッガ8を使用する場合は、ソースプログラム4をコンパイラ5に読み込ませ、コンパイル処理を実行させ、オブジェクトコード6を生成させる（ステップ31、32）。

【0023】同時に、コンパイラ5の最適化処理過程においてオブジェクトコード6のアドレス6aと、ソースプログラム4の行番号4aとの対応関係を示すシンボル情報7を作成させる。

【0024】一方、最適化処理を行わない旨の指示があった場合、最適化処理が抑止された状態でコンパイル処理が実行される（ステップ33）。

【0025】これにより、例えば図2の例のソースプログラム4においては、最適化処理の指示があった場合、ソースプログラム4の行番号4aの19行目から20行目はオブジェクトコード6のアドレス「85E8」から「85F4」、行番号4aの21行目から24行目はアドレス「85F8」から「8630」に対応するという具合に、複数行を一括して最適化処理が行われる。

【0026】次に、ソフトウェアデバッガ8に対して起動コマンドが入力され、ソフトウェアデバッガ8が起動されると（ステップ34）、ソフトウェアデバッガ8は検証対象となるソースプログラム4、そのオブジェクトコード6を読み込むと共に、シンボル情報7を読み込む（ステップ35）。

【0027】次に、ソフトウェアデバッガ8はソースプログラム4の表示コマンド（ソース表示指示）がキーボード2から入力されているかどうかを判定し、入力されていれば、ソフトウェアデバッガ8のソースプログラム表示処理8aは、図2に示すように、ソースプログラム4の各行に行番号4aを付加してディスプレイ1に表示する（ステップ36、37）。

【0028】その際、シンボル情報7を検索し、シンボル情報7の中に行番号4aと一致するものがある場合は、行番号4aに記号4b（例えば“#（シャープ）”記号）を付加して表示し、アドレス情報6aが割り当てられない他の行と区別して認識可能なように表示する。

【0029】この表示に対し、ソースプログラム4を最終ステップ以前の任意のステップで実行を一時停止させ、その段階における実行結果を検証するために、実行停止ステップに対応するソースプログラム4の番号4aがキーボード2からブレークポイントとして設定され、かつ実行開始指示が入力されたならば、ソフトウェアデ

する(ステップ38, 39)。

【0030】ソフトウェアデバッガ8のブレークポイント処理8d1は、キーボード2から予め設定されたブレークポイント(有効行番号)をシンボル情報7を参照して対応するアドレス情報に変換し、オブジェクトコード6の実行アドレス情報6aとが一致するかどうかを監視しているが、両者が一致した場合(ステップ41)、そのアドレス情報6aで示されるオブジェクトコードを実行した段階でデバッグ処理を一時停止させる(ステップ41)。

【0031】しかし、一致しない場合(すなわち実行アドレスがブレークポイントでない場合)は、最終アドレスかどうかを判定し(ステップ42)、最終アドレスであれば全てのデバッグ作業を終了させる。

【0032】例えば、ソフトウェアデバッガ8で行番号4aによる実行停止操作あるいは実行開始操作を行う場合、図2の例のソースプログラム4の19行目、21行目等の先頭行はソースプログラム4とアドレス情報6aとの対応がとれるので、実行停止操作等を有効なものとしてすることができるが、22行目から24行目までは対応するオブジェクトコード6のアドレス情報6aが不明となるため、これらの行に対する実行停止操作等は無効と扱われてしまう。

【0033】そこで、上記のように、オブジェクトコード6のアドレス6aが割り当てられているソースプログラム4の行番号4a、すなわち有効行を記号4bで明示し、この有効行をブレークポイントとして指定させるようにする。

【0034】これによって、コンパイル時に最適化処理を実行し、ソースプログラム行とオブジェクトコードのアドレスとが1対1に対応しなくなる部分が生じたとしても、オブジェクトコードのアドレスが割り当てられたソースプログラム行が明確になるので、テスト担当者は有効行のみを指定して実行停止指示または実行開始指示を行うことが可能になる。

【0035】ここで、ステップ38においてブレークポイントを設定した場合、テスト担当者がアドレス情報6aを割り当てられたソースプログラム行(有効行)を必ず指定するとは限らない。また、ソースプログラム表示処理8aは、図5に示すようにオブジェクトコード6のみをディスプレイ1に表示させるモードを備えているが、このモードにおいては有効行を正確に指示することはできない。

【0036】そこで、本実施例にあつては、テスト担当者によってブレークポイントが設定された場合、それが有効行に対応しているのかどうかを行番号判別処理8bにおいてシンボル情報7との照合によって判別し、有効行に対応している場合にはそのままブレークポイント処理8d1に引き渡すが、無効行に対応していた場合は、行番号補正処理8cに引渡し、ここで直前の有効行を示

6

す番号に補正した後にブレークポイント処理8d1に引き渡すようになっている。

【0037】例えば、図2の例において、ブレークポイントとして行番号「22」が設定された場合は、これを行番号「21」に補正してブレークポイント処理8d1に引き渡す。

【0038】これにより、ブレークポイント処理8d1では、ブレークポイントとして設定された行番号をシンボル情報7によって対応するアドレス情報6aに変換し、このアドレス情報6aで示されるアドレスが実行された段階でデバッグ作業を一時停止させることになる。

【0039】この場合、行番号を補正した時には直前の有効行で一時停止させることを警告メッセージや警告音でテスト担当者に通知する。

【0040】なお、デバッグ途中で一時停止させる場合に限らず、途中から再開する場合についても同様に実行させることができる。

【0041】従って、本実施例によれば、コンパイル時に最適化処理を実行し、ソースプログラム行とオブジェクトコードのアドレスとが1対1に対応しなくなる部分が生じたとしても、オブジェクトコードのアドレスが割り当てられたソースプログラム行が明確になるので、テスト担当者は有効行のみを指定して実行停止指示または実行開始指示を行うことが可能になる。

【0042】この結果、最適化処理を抑止することなく、すなわち実装置に組み込むプログラムと同じ状態で検証対象プログラムを検証することができる。また、有効行を予測する必要がなくなるので、1回の指示操作のみで有効行での実行停止または実行開始を実現することができる。

【0043】また、アドレスが割り当てられないソースプログラム行での実行停止指示または実行開始指示に対しては、オブジェクトコードアドレスが割り当てられている直前のソースプログラムの行番号に補正した後、オブジェクトコードに従う処理を実行させているので、テスト担当者が無効行を指定した場合でも、その操作が無効にならず、確実に一時停止させることができ、作業効率の低下を防止できる。

【0044】なお、実施例においては、有効行と無効行とを#記号によって区別しているが、これに限らず、輝度、色などを変える方法によって区別するようにしてもよい。

【0045】また、ソースプログラム4の表示形態としては、図4に示すように、最適化処理単位でソースプログラムの内容と対応するオブジェクトコードとを1対1にして表示させるようにしてもよい。

【0046】

【発明の効果】以上のように本発明によれば、コンパイル時に最適化処理を実行し、ソースプログラム行とオブジェクトコードのアドレスとが1対1に対応しなくなる

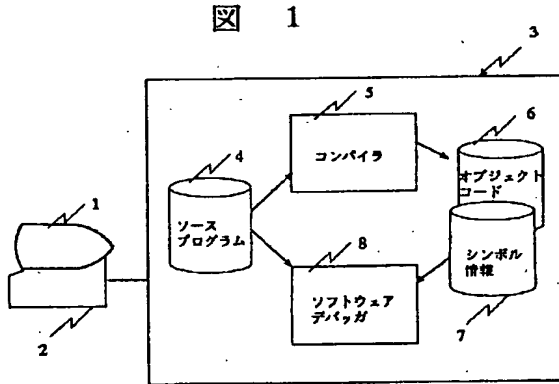
7

部分が生じたとしても、オブジェクトコードのアドレスが割り当てられたソースプログラム行が明確になるので、テスト担当者は有効行のみを指定して実行停止指示または実行開始指示を行うことが可能になる。

【0047】この結果、最適化処理を抑止することなく、すなわち実装置に組み込むプログラムと同じ状態で検証対象プログラムを検証することができる。また、有効行を予測する必要がなくなるので、1回の指示操作のみで有効行での実行停止または実行開始を実現することができ、作業効率を向上させることができる。

【0048】また、アドレスが割り当てられないソースプログラム行での実行停止指示または実行開始指示に対しては、オブジェクトコードアドレスが割り当てられている直前のソースプログラムの行番号に補正した後、オブジェクトコードに従う処理を実行させているので、テスト担当者が無効行を指定した場合でも、その操作が無効にならず、確実に一時停止させることができ、作業効

【図1】



8

率の低下を防止できる。

【図面の簡単な説明】

【図1】本発明を実施する装置の一実施例を示すブロック構成図である。

【図2】図1の詳細構成を示す図である。

【図3】実施例におけるデバッグ手順を示すフローチャートである。

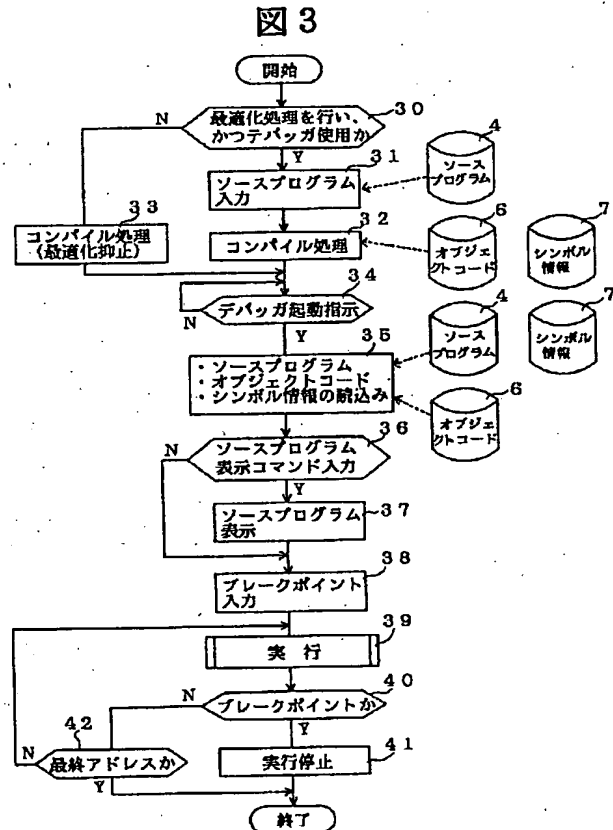
【図4】ソースプログラムの表示形式の他の例を示す図である。

10 【図5】オブジェクトコードのみの表示例を示す図である。

【符号の説明】

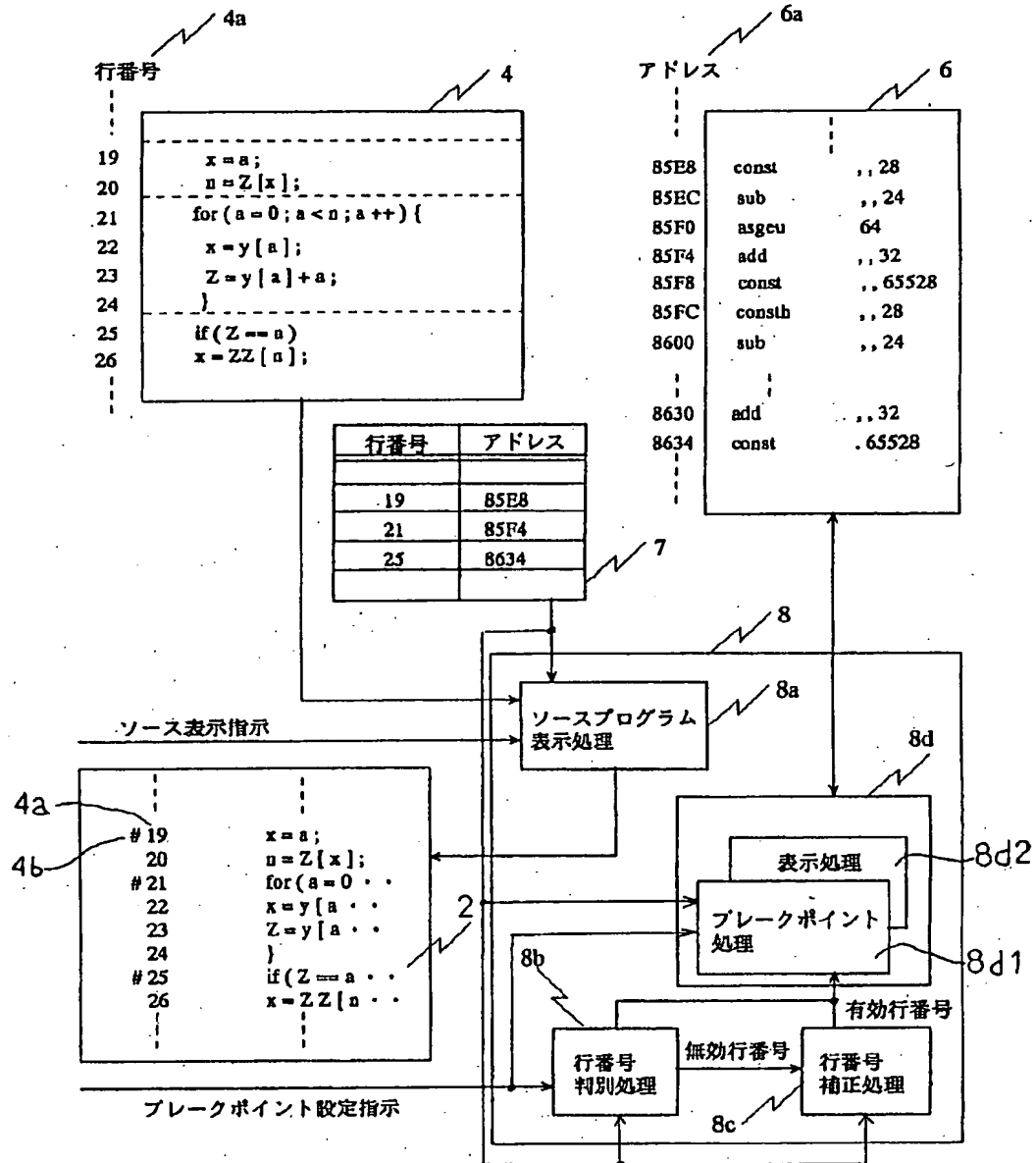
1…ディスプレイ、2…キーボード、3…ワークステーション、4…ソースプログラム、5…コンパイラ、6…オブジェクトコード、7…シンボル情報、8…ソフトウェアデバッガ、8a…ソースプログラム表示処理、8b…行番号判別処理、8c…行番号補正処理。

【図3】



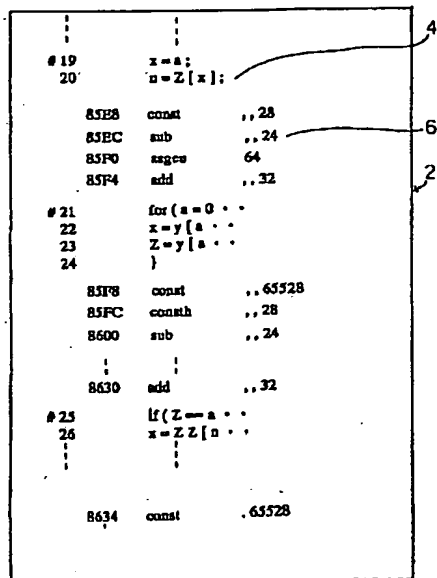
【図2】

図 2



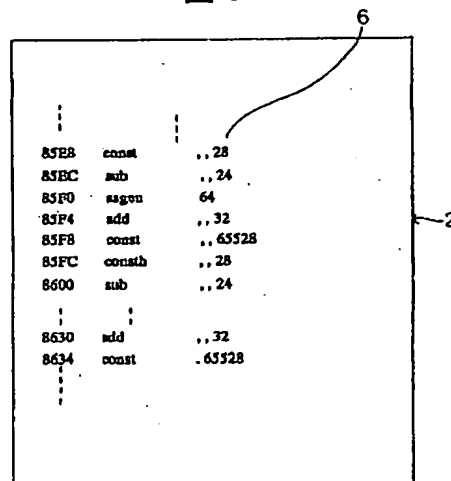
【図4】

図4



【図5】

図5



フロントページの続き

(72)発明者 佐々木 保
 神奈川県秦野市堀山下1番地 株式会社日立製作所汎用コンピュータ事業部内

(72)発明者 小泉 昭典
 神奈川県秦野市堀山下1番地 日立コンピュータエンジニアリング株式会社内